**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное автономное образовательное**

**учреждение высшего образования**

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(РУТ(МИИТ)**

Кафедра «Вычислительные системы, сети и информационная безопасность»

**Отчет По Дисциплине**

**«Алгоритмизация и программирование»**

**Лабораторная работа №3**

*Направление:* 10.03.01*Информационная безопасность*

*Профиль:**Безопасность компьютерных систем*

Выполнил:  
студент группы УИБ-112

Орлов Андрей Витальевич

Проверил:

Старший преподаватель Никольская Марина Николаевна

(должность, ФИО)

Старший преподаватель Цыганова Наталия Алексеевна

(должность, ФИО)

Москва 2021 г.

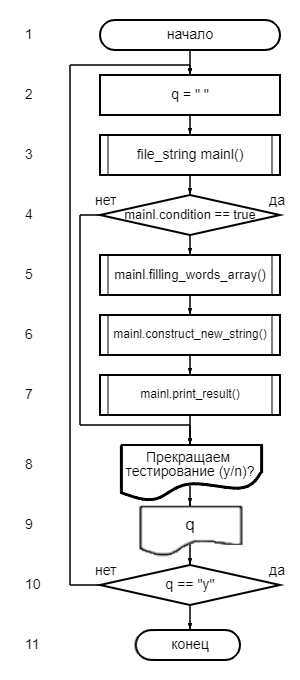
**Задание №3**

Задание: Задан текстовый файл f1. Удалить из него все однобуквенные слова и “лишние” пробелы. Результат записать в файл f2. Использовать функции, а также меню для выбора существующего или вновь создаваемого файла.

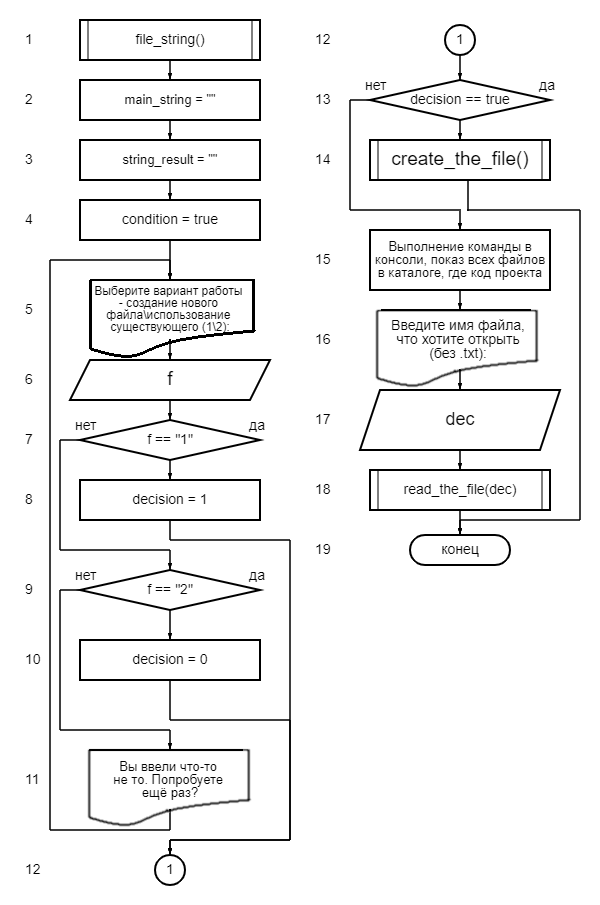
1. Таблица имён:

|  |  |  |
| --- | --- | --- |
| Исходные данные | | |
| f | строковый | Переменная для выбора действия программы – создание нового файла, или использование уже готового. |
| dec | строковый | Переменная, в которую вводится имя файла без расширения, который необходимо открыть. |
| s | строковый | Переменная для создания имени файла обычного текстового вида (\*.txt). |
| s22 | строковый | Переменная, в которую записывается строчка, которая затем будет записана в файл для дальнейшей её обработки пользователем |
| Рабочие переменные | | |
| main\_string | строковый | Переменная для считывания строки из файла |
| mainl | экземпляр класса file\_string | Объект для работы с файлами и строками |
| amount\_of\_spaces | целочисленный | Количество пробелов в исходной строке |
| main\_words | указатель строкового типа | Массив со всеми словами в строке |
| condition | логический | Переменная для проверки, есть ли с чем работать программе далее, или же процесс необходимо срочно прервать (в условиях) |
| decision | логический | Переменная для тернарного оператора – выбор режима работы программы – запись или чтение файлов. |
| helper | строковый | Переменная для потоковой считки текста из файла – обновляется при каждом считывании |
| a | переменная чтения файла | Переменная потоковой передачи информации из файла, в данной программе - в строки. |
| i | целочисленный | Переменная-счётчик в циклах |
| k | целочисленный | Переменная-счётчик в циклах |
| q | строковый | Переменная решения для тестировщика |
| Результат | | |
| string\_result | строковый | Строчка с удалёнными пробелами и однобуквенными словами. |
| f2 | переменная записи файла | Переменная, осуществляющая потоковую передачу строкового ответа в файл |

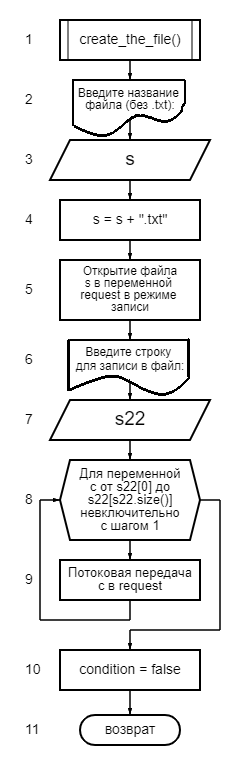
1. Блок-схема:



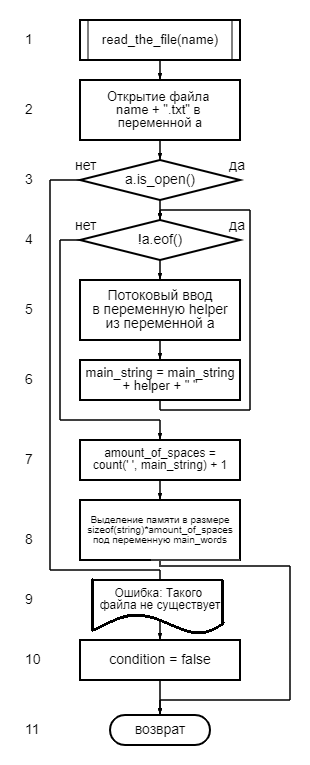
Блок-схема №1 – Основной код программы



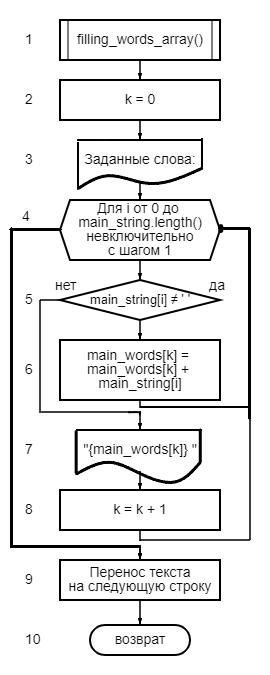
Блок-схема 2 (а) – Конструктор класса file\_string



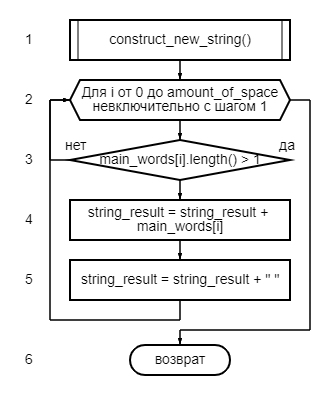
Блок-схема №3 (б) – Метод создания файла



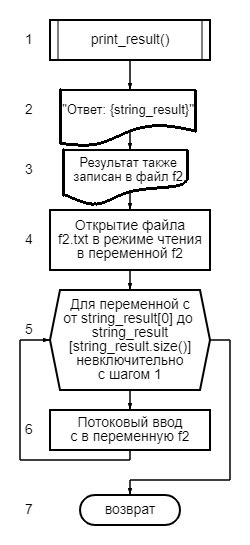
Блок-схема №4 (в) – Метод чтения файла



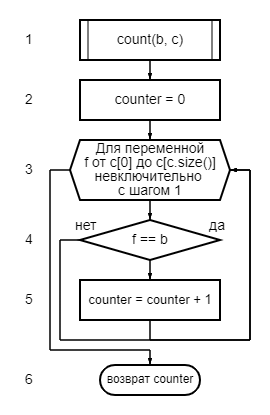
Блок-схема №5 (г) – Метод заполнения массива слов

\

Блок-схема №6 (д) – Метод конструирования новой строки



Блок-схема №7 (е)– Метод печати результата в файл и в консоль

1

Блок-схема №8 (ж)– Функция подсчёта количества символов в строке

1. Отладочный пример:

1 вариант.

1) Начало  
2) Инициализация переменной: q = “”  
3) Вызов конструктора класса file\_string для mainl  
Переход к блок-схеме 2 (а)  
2a) Инициализация переменной main\_string = “”  
3a) Инициализация переменной string\_result = “”  
4a) Инициализация переменной condition = true  
5а) Вывод на экран – Выберите вариант работы – создание нового файла\использование существующего (1\2)  
6а) Ввод переменной f = “1”  
7а) Условие – “1” == “1”, да, идём к блоку 8а  
8а) Переприсваивание переменных decision = 1  
12а) Ссылка на блок 13  
13а) Условие – 1 == true, да, переход к блоку 14а  
14а) Вызов метода create\_the\_file для экземпляра класса file\_string mainl,  
переход к блок-схеме 3 (б)  
2б) Вывод на экран – Введите название файла (без .txt):  
3b) Ввод s = “f1”  
4b) Переприсваивание переменных s = “f1.txt”  
5b) Открытие файла f1.txt в переменной request в режиме записи  
6b) Вывод – Введите строку для записи в файл:  
7b) Ввод s22 = “Мы русские, с нами бог!”  
8b-9b) Потоковая передача s22 в файл через потоковую переменную request  
10b) Переприсваивание переменных: condition = false  
11b) Возврат из метода в конструктор класса  
19) Конец конструктора классов, переход в основную часть программы   
4) Условие false == true, нет, переход к блоку 8  
8) Вывод – Прекращаем тестирование? (y\n)  
9) Ввод q = ‘n’  
10) Условие – ‘y’ == ‘n’, нет, идём к блоку 2  
2) Переприсваивание переменных q = “ “  
3) Вызов конструктора класса file\_string для mainl  
Переход к блок-схеме 2 (а)  
2a) Инициализация переменной main\_string = “”  
3a) Инициализация переменной string\_result = “”  
4a) Инициализация переменной condition = true  
5а) Вывод на экран – Выберите вариант работы – создание нового файла\использование существующего (1\2)  
6а) Ввод переменной f = “2”  
7а) Условие – “2” == “1”, нет, идём к блоку 9а  
9а) Условие – “2” == “2”, да, идём к блоку 10а  
10а) Переприсваивание переменной decision = 0  
12a) Ссылка на блоку 13а  
13а) Условие – 0 == true, нет, идём к блоку 15  
15а) Выполнение команды “DIR” в консоли, показ всех файлов в каталоге, где располагается код проекта.   
16а) Вывод – Введите имя файла, что хотите открыть (без .txt)  
17а) Ввод dec = “f1”  
18а) Запуск метода read\_the\_file(“f1”), переход к блоку-схеме №4 (в)  
2в) Открытие файла “f1.txt” в переменной а  
3в) Файл открыт – да, переход к блоку 4в  
4-5-6в) Потоковый ввод текста из файла через переменную а в строковый тип данных main\_string через helper для избежания перезаписи строки.  
После выполнения main\_string = “Мы русские, с нами бог!”  
7в) Переприсваивание переменных amount\_of\_space = count(‘ ‘, “Мы русские, с нами бог!”), функция считает пробелы, так что amount\_of\_space = 4.  
8в) Выделение памяти под main\_words в размере sizeof(string)\*4  
11в) Возврат в конструктор класса  
19а) Возврат в основную часть программы  
4) Условие – true == true, да, переход к блоку 5  
5) Запуск метода filling\_words\_array() для экз. кл. mainl (б-с №5 (г))  
2г) Инициализация переменной k = 0  
3г) Вывод – Заданные слова:  
4-5-6-7-8г) Последовательное считывание слов из main\_string в main\_words, считываем ровно amount\_of\_space + 1 слов.  
После выполнения этих пунктов получаем   
main\_words = {“Мы” “русские,” “с”, “нами”, “бог!”}  
9г) Перенос текста на следующую строку  
10г) Возврат в основную часть программы  
 6) Запуск метода construct\_new\_string() для экз. кл. mainl (б-с №6 (д))  
2-3-4-5д) Если у элемента массива main\_words длина превышает 1, тогда прибавляем к нашему результату это слово. А затем добавляем пробел после него. Результат – string\_result = “Мы русские, нами бог!”  
6д) Возврат в основную часть программы  
7) Запуск метода print\_result() для экз. кл. mainl (б-с №7 (е))  
2е) Вывод – Ответ: Мы русские, нами бог!  
3e) Вывод – Результат также записан в файл f2.  
4-5-6e) Потоковый ввод string\_result в f2  
Возврат в основную часть программы  
8) Вывод – Прекращаем тестирование? (y\n)?  
9) Ввод q = ‘y’  
10) Условие - ‘y’ == ‘y’, да, переход к блоку 11  
11) Конец

1. Код программы:

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

// Функция для подсчёта количества символов в заданной строке

int count(char b, string c) {

int counter = 0;

for (auto f : c)

if (f == b)

counter++;

return counter;

}

// Класс для обработки строк, что хранятся в файлах

class file\_string {

private:

string main\_string = "";

string string\_result = "";

int amount\_of\_spaces;

string\* main\_words;

public:

bool condition = true;

// Конструктор класса, отвечающий за создание\чтение файлов.

file\_string() {

bool decision; string f;

while (true) {

cout << "Выберите вариант работы - создание нового файла\\использование существующего (1\\2): ";

getline(cin, f);

if (f == "1") {

decision = 1;

break;

}

else if (f == "2") {

decision = 0;

break;

}

else

cout << "Вы ввели что-то не то. Попробуете ещё раз?" << endl;

}

switch (decision == true) {

case 1:

create\_the\_file();

break;

case 0:

system("dir \*.txt");

cout << "Введите имя файла, что хотите открыть (без .txt): ";

string dec; getline(cin, dec);

read\_the\_file(dec);

break;

}

}

// Метод для обработки случая создания файла пользователем

void create\_the\_file() {

cout << "Введите название файла (без .txt): ";

string s; getline(cin, s); s += ".txt";

ofstream request(s); string s22;

cout << "Введите строку для записи в файл: ";

getline(cin, s22);

for (auto c : s22) {

request << c;

}

condition = false;

}

// Метод для обработки случая чтения файла пользователем

void read\_the\_file(string name) {

string helper;

ifstream a(name + ".txt");

if (a.is\_open()) {

while (!a.eof()) {

a >> helper;

main\_string += helper + " ";

}

amount\_of\_spaces = count(' ', main\_string) + 1;

main\_words = new string[amount\_of\_spaces];

}

else {

cout << "Ошибка: Такого файла не существует" << endl;

condition = false;

}

}

// Метод для выделения подходящих по условию слов в массив

void filling\_words\_array() {

int k = 0; cout << "Заданные слова: ";

for (int i = 0; i < main\_string.length(); i++) {

if (main\_string[i] != ' ')

main\_words[k] += main\_string[i];

else {

cout << main\_words[k] << " ";

k++;

}

}

cout << endl;

}

// Строим новую строчку, не включая пробелы и однобуквенные слова

void construct\_new\_string() {

for (int i = 0; i < amount\_of\_spaces; i++) {

if (main\_words[i].length() > 1) {

string\_result += main\_words[i];

string\_result += " ";

}

}

}

// Выводим строчку, которая удовлетворяет ответу.

void print\_result() {

cout << "Ответ: " << string\_result << endl;

cout << "Результат также записан в файл f2.\n";

ofstream f2("f2.txt");

for (auto c : string\_result)

f2 << c;

}

};

int main() {

// Подключаем русский язык в консоли

setlocale(LC\_ALL, "rus");

system("chcp 1251 >> null");

// Для удобства тестирования программы

while (true) {

string q = "";

file\_string mainl;

if (mainl.condition == true) {

mainl.filling\_words\_array();

mainl.construct\_new\_string();

mainl.print\_result();

}

cout << "Прекращаем тестирование? (y/n): ";

getline(cin, q);

if (q == "y")

break;

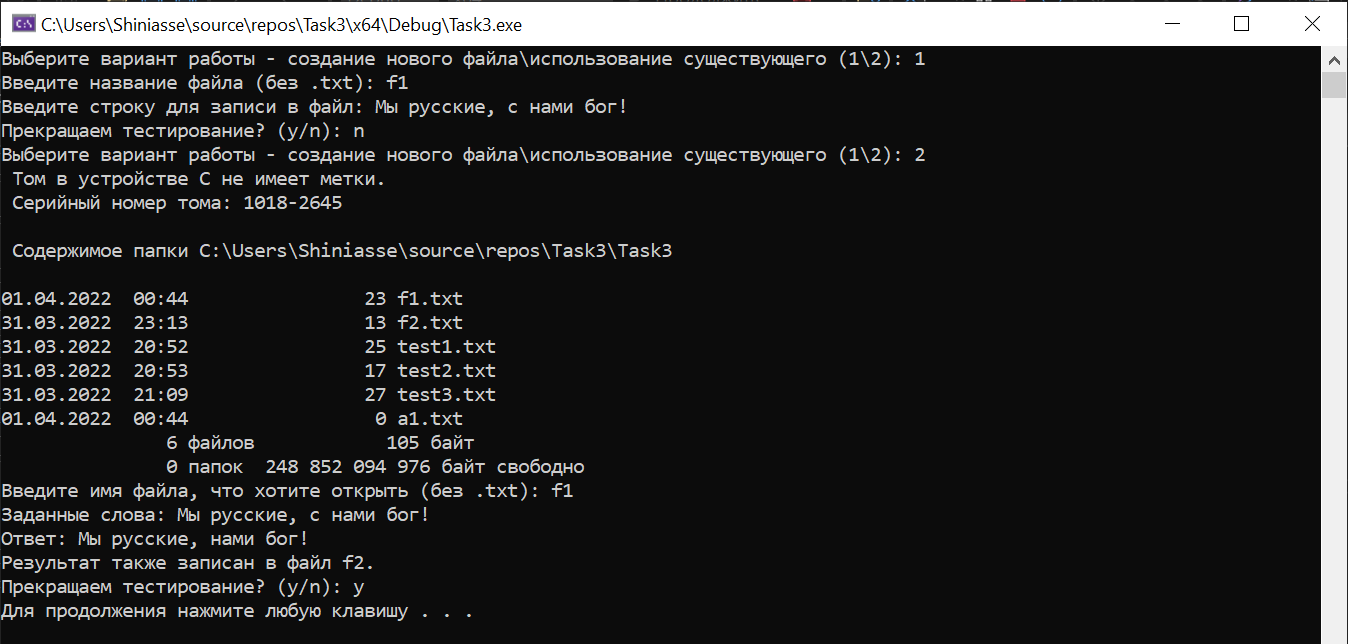
}

system("pause");

return 0;

}

1. Результат выполнения работы программы:



Отладка №1

1. Вывод:

В ходе выполнения работы были изучены способы связи различных библиотек с работой классов – в частности, затрагивали тему библиотеки файловых потоков fstream. С помощью неё было реализовано меню для выбора файлов по названиям (показ имён файлов был осуществлен через консольную команду “dir”). Также была отлажена работа закрытых и открытых полей класса, также была связана функция с работой классового метода (подсчёт пробелов в строке).

На контрольных примерах мы убедились, что код работает корректно.  
Был оформлен комплект документации на программный код.